View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Dagstuhl Research Online Publication Server

On the Average-Case Complexity of MCSP and Its Variants*

Shuichi Hirahara[1] **and Rahul Santhanam**[2]

1 **Department of Computer Science, The University of Tokyo, Tokyo, Japan**
hirahara@is.s.u-tokyo.ac.jp
2 **Department of Computer Science, University of Oxford, Oxford, UK**
rahul.santhanam@cs.ox.ac.uk

Abstract

We prove various results on the complexity of MCSP (Minimum Circuit Size Problem) and the related MKTP (Minimum Kolmogorov Time-Bounded Complexity Problem):

- We observe that under standard cryptographic assumptions, MCSP has a *pseudorandom self-reduction*. This is a new notion we define by relaxing the notion of a random self-reduction to allow queries to be pseudorandom rather than uniformly random. As a consequence we derive a weak form of a worst-case to average-case reduction for (a promise version of) MCSP. Our result also distinguishes MCSP from natural NP-complete problems, which are not known to have worst-case to average-case reductions. Indeed, it is known that strong forms of worst-case to average-case reductions for NP-complete problems collapse the Polynomial Hierarchy.
- We prove the first non-trivial formula size lower bounds for MCSP by showing that MCSP requires nearly quadratic-size De Morgan formulas.
- We show average-case superpolynomial size lower bounds for MKTP against $\mathsf{AC}^0[p]$ for any prime p.
- We show the hardness of MKTP on average under assumptions that have been used in much recent work, such as Feige's assumptions, Alekhnovich's assumption and the Planted Clique conjecture. In addition, MCSP is hard under Alekhnovich's assumption. Using a version of Feige's assumption against co-nondeterministic algorithms that has been conjectured recently, we provide evidence for the first time that MKTP is not in coNP. Our results suggest that it might worthwhile to focus on the *average-case hardness* of MKTP and MCSP when approaching the question of whether these problems are NP-hard.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases minimum circuit size problem, average-case complexity, circuit lower bounds, time-bounded Kolmogorov complexity, hardness

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.7

1 Introduction

Progress in complexity theory is often correlated with an improved understanding of *meta-computational problems*, i.e., problems that themselves encode questions about circuits or algorithms. Consider the canonical meta-computational problem Circuit-SAT, which asks whether a given circuit has a satisfying assignment. Results on the complexity of the

* The second author was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant No. 615075. Part of this work was done during a visit of the first author to Oxford supported by the second author's ERC grant.


© Shuichi Hirahara and Rahul Santhanam;
licensed under Creative Commons License CC-BY
32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 7; pp. 7:1–7:20


Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Circuit-SAT problem and its instantiations for restricted classes of circuits such as CNFs have had a major impact on complexity theory. These include the Cook-Levin theorem showing that CNF-SAT is NP-complete, the PCP theorem showing that CNF-SAT is hard to approximate, and more recently the connection established by Ryan Williams [35] between "non-trivial" algorithms for Circuit-SAT and circuit lower bounds, which has led to a new *algorithmic paradigm* for proving new circuit lower bounds.

A meta-computational problem that is in a sense dual to Circuit-SAT is MCSP: given the truth table of a Boolean function f and a parameter s, determine if f has circuits of size at most s. While Circuit-SAT asks about a property of the Boolean function encoded by a given circuit, MCSP asks about whether an explicitly given Boolean function can be encoded by a small circuit. It is easy to see that MCSP, like Circuit-SAT, is in NP; however, its precise complexity is much less well understood. This is despite the fact that MCSP was recognized as fundamental by theoretical computer scientists in the Soviet Union as early as the 1950s, as discussed in a fascinating survey by Trakhtenbrot [34]. In the more recent past, interest in MCSP was reawakened by a paper of Kabanets and Cai [24], building on the "Natural Proofs" work of Razborov and Rudich [30], and there have been several papers [5, 7, 10, 6, 9, 3, 27, 19, 18, 8] since on the complexity of the problem.

We do not have clear answers even to some basic questions about the complexity of MCSP. These questions include: Is MCSP NP-complete? Does the MCSP problem have similar structural properties to Circuit-SAT and other standard NP-complete problems, such as paddability and downward self-reducibility, or does it have properties such as random self-reducibility which are characteristic of problems such as Factoring and DiscreteLogarithm which are used in cryptography? What is the strongest evidence we can provide that MCSP is not in polynomial time? Are there formal connections between variants of MCSP which arise from using different circuit classes or fixing the parameter s in advance? Can we show unconditional complexity lower bounds for MCSP, for restricted classes of circuits such as sub-quadratic size formulas and sub-exponential size constant depth circuits with prime modular gates?

Our main argument in this paper is that it is valuable to look at MCSP through the lens of *average-case complexity*. By adopting this perspective, we are able to make progress on all of the questions above. We must first explain, however, what we mean by the average-case complexity of MCSP. Rather than studying MCSP itself, we study its parameterized version MCSP$[s]$, where the size function s is given in advance. We consider the complexity of this problem under the uniform distribution on inputs to the problem. Note that an input to the problem is simply the truth table of a Boolean function, so the distribution on inputs we consider corresponds to the uniform distribution on n-bit Boolean functions, which is fairly natural in this context. When the size function $s(n) = o(2^n/n)$, most Boolean functions do not have circuits of size s, and hence most inputs to the problem MCSP$[s]$ are NO inputs. Thus the problem is highly biased, and the algorithm just outputting NO on all instances has a very high probability of success. This means that it is uninteresting to consider a version of average-case complexity where the algorithm is allowed to make errors. Instead, we consider the standard zero-error notion, where the algorithm never outputs an incorrect answer, but is allowed to output '?' when it doesn't know the answer for an input.

Why do we believe studying the average-case complexity is more fruitful than studying the worst-case complexity? For one thing, it makes the theory cleaner. Let $\mathfrak{C}_1$ and $\mathfrak{C}_2$ be classes of circuits such that $\mathfrak{C}_1 \subseteq \mathfrak{C}_2$, and for a size function $s = o(2^n/n)$, let MCSP-$\mathfrak{C}_1[s]$ and MCSP-$\mathfrak{C}_2[s]$ be the variants of MCSP corresponding to the classes $\mathfrak{C}_1$ and $\mathfrak{C}_2$ respectively. Intuitively, it seems that MCSP-$\mathfrak{C}_2[s]$ must be at least as hard a problem as MCSP-$\mathfrak{C}_1[s]$

given that it concerns a more general class of circuits; however, in the setting of worst-case complexity, no formal connection between the two problems is known. In the setting of average-case complexity, in contrast, it is quite easy to show that the identity function is a reduction from the latter to the former. Similarly, given size functions s_1 and s_2 such that $s_1 \leq s_2$, it seems that $\mathsf{MCSP}[s_2]$ should be at least as hard as $\mathsf{MCSP}[s_1]$. Again, no formal reduction is known in the worst-case setting, but the identity function works as a reduction in the average-case setting. Thus the average-case setting seems to correspond more closely to our intuitions about the complexity of the problem.

A second point is that current techniques for proving results about the complexity of MCSP almost invariably yield results also on the average-case complexity. Most techniques we know go through some notion of pseudorandomness, and pseudorandomness is intrinsically an average-case notion. Thus, if we aim to prove results on the hardness of MCSP using current techniques, we should first aim for average-case hardness rather than worst-case hardness.

We now proceed to give a more detailed description of our results. In some cases, our results are not about MCSP but about a surrogate of it called MKTP. Rather than asking if the input has small circuits when interpreted as the truth table of a Boolean function, the MKTP problem asks if the input is compressible by a program from which individual bits of the input can be efficiently computed. Thus, while MCSP is a question about compression by circuits, MKTP is a question about compression by programs. For a formal definition, we refer to Section 2. Using known strong relationships between uniform and non-uniform complexity, MCSP and MKTP are closely related, and until recently all known results about one problem also applied to the other. A recent exception is [3], and our paper is another exception - it seems that hardness results are occasionally easier to show for MKTP because it corresponds to a more fine-grained notion of compressibility than MCSP.

1.1 Our Results

We first investigate the possibility of worst-case to average-case connections for MCSP. It is known that nonadaptive worst-case to average-case connections for NP-complete problems collapse the Polynomial Hierarchy [12]. Hence, if we could show a worst-case to average-case connection for MCSP, it would give strong evidence against the NP-hardness of MCSP. We are not able to do this; however, under standard cryptographic assumptions, we give a *pseudorandom self-reduction* for a promise version of MCSP. Recall that a random self-reduction is a reduction from a computational problem to itself where the queries are uniformly distributed and of the same length as the input. A random self-reduction gives a strong worst-case to average-case connection for a problem. Our notion of pseudorandom self-reduction relaxes the original notion by allowing the queries to be pseudorandomly distributed rather than randomly distributed. While our result does not give evidence that MCSP is not NP-complete, it does distinguish the MCSP problem from natural NP-complete problems, for which pseudorandom self-reductions are unknown even under standard cryptographic assumptions, to the best of our knowledge.

▶ **Theorem 1** (Pseudorandom self-reductions: Informal Version)**.** *Suppose exponentially hard one-way functions exist. Let s be a size function such that $s(n) = n^{\omega(1)}$ and $s(n) = o(2^n/n)$. There is a constant c such that there is a pseudorandom self-reduction for the promise version of* MCSP*, where the YES instances are truth tables of Boolean functions of circuit complexity at most $s(n) - n^c$ and the NO instances are truth tables of Boolean functions of circuit complexity at least $s(n) + n^c$.*

Though pseudorandom self-reductions do not give worst-case to average-case reductions in full generality as random self-reductions, they do give a weak version of such reductions, as described in more detail in Section 3. The proof idea we use to establish Theorem 1 is a twist on the idea used by [30] to rule out natural proofs under cryptographic assumptions.

Next we attempt to prove unconditional lower bounds for MCSP, and MKTP. We show such lower bounds in two settings where lower bounds were unknown. The first is a lower bound for MCSP against subquadratic De Morgan formulae, and the second is an average-case lower bound for MKTP against polynomial-size constant depth circuits with Mod p gates, for prime p. Both proofs exploit connections with pseudorandom generators, in the first case the pseudorandom generators of [20] against formulas, and in the second case the pseudorandom generators of [14] against $\mathsf{AC}^0[p]$ using resamplability. Note that in both settings traditional lower bound techniques such as the method of random restrictions, the Neciporuk technique and the polynomial method do not appear to be directly applicable.

▶ **Theorem 2** (Unconditional lower bounds: Informal Version)**.** *There are size functions s and s' such that* MCSP$[s]$ *does not have De Morgan formulae of size $N^{2-\Omega(1)}$, and* MKTP$[s']$ *cannot be decided with $\Omega(1)$ success on average by polynomial-size constant-depth circuits with Mod p gates, where p is any prime.*

Finally, and perhaps most strikingly, we show the hardness of MKTP on average under various assumptions that have been intensively studied recently, such as Feige's hypothesis [15], Alekhnovich's hypothesis [2] and the Planted Clique conjecture [23, 26]. These are the first hardness results for MCSP or MKTP under assumptions for problems that are not known to be in SZK. The fact that MKTP is hard on average under average-case hardness assumptions about NP-complete problems such as SAT and Clique might be taken as providing mild evidence in favour of the problem being NP-hard. Also, it has been conjectured by Ryan O'Donnell (personal communication) that Feige's hypothesis holds even with respect to co-nondeterministic polynomial time algorithms, and under this conjecture MKTP is not in coNP. We note that [5] observe that MKTP is not in coNP under a conjecture of Rudich [31], but the conjecture of O'Donnell is in our opinion more natural and plausible, relating as it does to Random Satisfiability, which is a problem that has been well studied algorithmically.

▶ **Theorem 3** (Hardness on Average under Popular Conjectures, Informal Version)**.** MKTP *is hard on average assuming Feige's hypothesis, Alekhnovich's hypothesis or the Planted Clique conjecture.* MCSP *is hard on average assuming Alekhnovich's hypothesis.*

2 Preliminaries and Notation

2.1 Boolean Function Complexity

We use $\mathcal{F}_m$ to denote the set of all Boolean functions $f\colon \{0,1\}^m \to \{0,1\}$. If W is a probability distribution, we use $w \sim W$ to denote an element sampled according to W. Similarly, for a finite set A, we use $a \sim A$ to denote that a is selected uniformly at random from A. Under this notation, $f \in \mathcal{F}_m$ represents a fixed function, while $f \sim \mathcal{F}_m$ is a uniformly random function. For convenience, we let $\mathcal{U}_n \stackrel{\text{def}}{=} \{0,1\}^n$. Following standard notation, $X \equiv Y$ denotes that random variables X and Y have the same distribution. We use standard asymptotic notation such as $o(\cdot)$ and $O(\cdot)$, and it always refer to a parameter $n \to \infty$, unless stated otherwise.

We say that $f, g \in \mathcal{F}_n$ are ε-close if $\Pr_{x\sim\mathcal{U}_n}[f(x) = g(x)] \geq 1 - \varepsilon$. We say that $h \in \mathcal{F}_n$ computes f with advantage δ if $\Pr_{x\sim\mathcal{U}_n}[f(x) = h(x)] \geq 1/2 + \delta$. It will sometimes be convenient to view a Boolean function $f \in \mathcal{F}_m$ as a subset of $\{0,1\}^m$ in the natural way.

We often represent Boolean functions as strings via the truth table mapping. Given a Boolean function $f \in \mathcal{F}_n$, $\mathtt{tt}(f)$ is the 2^n-bit string which represents the truth table of f in the standard way, and conversely, given a string $y \in \{0,1\}^{2^n}$, $\mathtt{fn}(y)$ is the Boolean function in $\mathcal{F}_n$ whose truth table is represented by y.

We identify each Boolean function $f : \{0,1\}^* \to \{0,1\}$ with a language $L \subseteq \{0,1\}^*$, where for any $x \in \{0,1\}^*$, $x \in L$ iff $f(x) = 1$. A promise problem is a pair (Π_{YES}, Π_{NO}) of languages over $\{0,1\}$, such that $\Pi_{YES} \cap \Pi_{NO} = \emptyset$. We say a language $L \subseteq \{0,1\}^*$ is consistent with a promise problem (Π_{YES}, Π_{NO}) if $\Pi_{YES} \subseteq L$ and $\Pi_{NO} \subseteq \bar{L}$.

Let $\mathfrak{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ be a class of Boolean functions, where each $\mathcal{C}_n \subseteq \mathcal{F}_n$. Given a language $L \subseteq \{0,1\}^*$, we write $L \in \mathfrak{C}$ if for every large enough n we have that $L_n \stackrel{\text{def}}{=} \{0,1\}^n \cap L$ is in $\mathcal{C}_n$. Often we will abuse notation and view $\mathfrak{C}$ as a class of Boolean circuits. For convenience, we use number of wires to measure circuit size. We denote by $\mathfrak{C}[s(n)]$ the set of n-variable $\mathfrak{C}$-circuits of size at most $s(n)$. As usual, we say that a uniform complexity class Γ is contained in $\mathfrak{C}[\mathsf{poly}(n)]$ if for every $L \in \Gamma$ there exists $k \geq 1$ such that $L \in \mathfrak{C}[n^k]$.

Given a sequence of Boolean functions $\{f_n\}_{n\in\mathbb{N}}$ with $f_n \colon \{0,1\}^n \to \{0,1\}$, we let $\mathfrak{C}^f$ denote the extension of $\mathfrak{C}$ that allows $\mathcal{C}_n$-circuits to have oracle gates computing f_n.

We will use a few other standard notions, and we refer to standard textbooks in computational complexity and circuit complexity for more details.

2.2 Natural Proofs and the Minimum Circuit Size Problem

We say that $\mathfrak{R} = \{\mathcal{R}_n\}_{n\in\mathbb{N}}$ is a *combinatorial property* (of Boolean functions) if $\mathcal{R}_n \subseteq \mathcal{F}_n$ for all n. We use $L_{\mathfrak{R}}$ to denote the language of truth-tables of functions in $\mathfrak{R}$. Formally, $L_{\mathfrak{R}} = \{y \mid y = \mathtt{tt}(f) \text{ for some } f \in \mathcal{R}_n \text{ and } n \in \mathbb{N}\}$.

▶ **Definition 4** (Natural Properties [30]). Let $\mathfrak{R} = \{\mathcal{R}_n\}$ be a combinatorial property, $\mathfrak{C}$ a circuit class, and $\mathfrak{D}$ a (uniform or non-uniform) complexity class. We say that $\mathfrak{R}$ is a $\mathfrak{D}$-natural property useful against $\mathfrak{C}[s(n)]$ if there is $n_0 \in \mathbb{N}$ such that the following holds:

(i) *Constructivity.* $L_{\mathfrak{R}} \in \mathfrak{D}$.
(ii) *Density.* For every $n \geq n_0$, $\Pr_{f\sim\mathcal{F}_n}[f \in \mathcal{R}_n] \geq 1/2^{O(n)}$.
(iii) *Usefulness.* For every $n \geq n_0$, we have $\mathcal{R}_n \cap \mathcal{C}_n[s(n)] = \emptyset$.

▶ **Definition 5** (Minimum Circuit Size Problem). Let $\mathfrak{C}$ be a circuit class. The Minimum Circuit Size Problem for $\mathfrak{C}$, abbreviated as MCSP-$\mathfrak{C}$, is defined as follows:

- *Input.* A pair (y, s), where $y \in \{0,1\}^{2^n}$ for some $n \in \mathbb{N}$, and $1 \leq s \leq 2^n$ is an integer (inputs not of this form are rejected).
- *Question.* Does $\mathtt{fn}(y)$ have $\mathfrak{C}$-circuits of size at most s?

We also define variants of this problem, where the circuit size is not part of the input.

▶ **Definition 6** (Parameterized Minimum Circuit Size Problem). Let $\mathfrak{C}$ be a circuit class, and $s \colon \mathbb{N} \to \mathbb{N}$ be a function. The Minimum Circuit Size Problem for $\mathfrak{C}$ with parameter s, abbreviated as MCSP-$\mathfrak{C}[s]$, is defined as follows:

- *Input.* A string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$ (inputs not of this form are rejected).
- *Question.* Does $\mathtt{fn}(y)$ have $\mathfrak{C}$-circuits of size at most $s(n)$?

▶ **Definition 7** (Parameterized Minimum Circuit Size Gap Problem). Let $\mathfrak{C}$ be a circuit class, and let $c, s \colon \mathbb{N} \to \mathbb{N}$ be functions such that $c(n) \geq s(n)$ for all $n \in \mathbb{N}$. The Minimum Circuit Size Gap Problem for $\mathfrak{C}$ with parameters c and s, abbreviated as MCSP-$\mathfrak{C}[c, s]$ is a promise problem defined as follows:

- *YES Instance.* Any string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$, such that $\mathtt{fn}(y)$ has $\mathfrak{C}$-circuits of size at most $s(n)$.
- *NO Instance.* Any string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$, such that $\mathtt{fn}(y)$ has no $\mathfrak{C}$-circuits of size at most $c(n)$.

When $\mathfrak{C}$ is not explicitly specified, we take $\mathfrak{C}$ to be the class of Boolean circuits.

Note that a dense property useful against $\mathfrak{C}[s(n)]$ is a dense subset of the complement of MCSP-$\mathfrak{C}[s]$.

2.3 Time-Bounded Kolmogorov Complexity and MKTP

KT-complexity was proposed in [5] in order to model circuit complexity in terms of time-bounded Kolmogorov complexity: it is known that $\mathrm{KT}(\mathtt{tt}(f))$ and the minimum circuit size of f are polynomially-related to each other. As usual, we fix a universal random-access Turing machine U that simulates all Turing machines efficiently. The KT-complexity of a string x is the minimum of $|d| + t$, where d is a string for describing x implicitly and t is the time it takes to output x. More formally, we have the definition below, where U^d denotes the Turing machine U with random access to the string d:

▶ **Definition 8** (KT-complexity [5])**.** Let $x = x_1 \cdots x_n \in \{0,1\}^n$. The KT-complexity of x is defined as follows.

$$\mathrm{KT}(x) := \min\{|d| + t \mid \ U^d(i) = x_i \text{ in } t \text{ steps for any } 1 \leq i \leq n+1 \ \}.$$

Here, x_{n+1} is defined as $\perp$ (a stop symbol).

For this complexity measure, a problem related to MCSP is defined as follows.

▶ **Definition 9** (Minimum Kolmogorov Time-bounded Complexity Problem)**.** The Minimum Kolmogorov Time-bounded Complexity Problem, abbreviated as MKTP, is defined as follows:

- *Input.* A pair (y, s), where $y \in \{0,1\}^*$ and $s \in \mathbb{N}$ (inputs not of this form are rejected).
- *Question.* $\mathrm{KT}(y) \leq s$?

▶ **Definition 10** (Parameterized Minimum Kolmogorov Time-bounded Complexity Problem)**.** Let $s\colon \mathbb{N} \to \mathbb{N}$ be a function. The Minimum Kolmogorov Time-bounded Complexity Problem with parameter s, abbreviated as MKTP$[s]$, is defined as follows:

- *Input.* A string $y \in \{0,1\}^*$.
- *Question.* $\mathrm{KT}(y) \leq s(|y|)$?

2.4 Average-Case Complexity

We require various notions of easiness on average. Let $\mathcal{D} = \{D_n\}, n \in \mathbb{N}$, be a sequence of distributions, where each D_n has support in $\{0,1\}^n$. A *distributional problem* is a pair $(L, \mathcal{D})$, where $L \subseteq \{0,1\}^*$ and $\mathcal{D}$ is a sequence of distributions.

▶ **Definition 11** (Easiness on Average)**.** Let $\mathfrak{C}$ be a (uniform or non-uniform) complexity class, and let $\varepsilon : \mathbb{N} \to [0,1]$ be a success parameter. We say a distributional problem $(L, \mathcal{D})$ is solvable in $\mathfrak{C}$ on average with success ε if there is a $\mathfrak{C}$-algorithm A such that for each $x \in \{0,1\}^*$, $A(x) = L(x)$ or $A(x) =' ?'$, and for each $n \in \mathbb{N}$, with probability at least $\varepsilon(n)$ over $x \sim D_n$, $A(x) = L(x)$. We say that a language L is in $\mathfrak{C}$ on average with success ε if (L, U_n) is in $\mathfrak{C}$ on average with success ε.

We observe that the easiness on average of certain variants of MCSP is equivalent to natural proofs, and moreover that the easiness on average is robust with regard to the success parameter.

▶ **Proposition 12** (Natural Proofs and Easiness of MCSP on Average). *Let $s : \mathbb{N} \to \mathbb{N}$ be a size function such that $s(n) = 2^n/n^{\omega(1)}$. The following are equivalent:*

1. *For all $c > 0$ there are* P*-natural (resp.* SIZE(poly)*-natural) properties useful against* SIZE$(s(cn))$.
2. *For all $c > 0$* MCSP$[s(cn)]$ *is solvable in* P *(resp.* SIZE(poly)*) on average with success* $1/\operatorname{poly}(N)$*, where $N = 2^n$ is the input size for the* MCSP *problem.*
3. *For all $c > 0$* MCSP$[s(cn)]$ *is solvable in* P *(resp.* SIZE(poly)*) on average with success probability* $1 - 1/\operatorname{poly}(N)$.

Proof. We provide just a sketch. Let $\mathfrak{C}$ be P or SIZE(poly). The proof is based on two observations. The first is that a $\mathfrak{C}$-natural property of density ε useful against $s(n)$-size Boolean circuits immediately yields that MCSP$[s]$ is in $\mathfrak{C}$ on average with success ε, simply by using the constructivity of the property and answering '?' on any input truth table that does not satisfy the property. Conversely, if MCSP$[s]$ is in $\mathfrak{C}$ on average with success ε, this implies a $\mathfrak{C}$-natural property with density $\varepsilon - 1/N^{\omega(1)}$, where a truth table is in the property iff the average-case algorithm answers 0 on the truth table. Since $s(n) = 2^n/n^{\omega(1)}$, the algorithm can only answer 1 on a $1/N^{\omega(1)}$ fraction of inputs, and hence the density of inputs on which the algorithm answers 0 is at least $\varepsilon - 1/N^{\omega(1)}$.

The second observation is that the density for natural properties can be amplified, with some cost to the usefulness. Given a natural property $\mathfrak{R}$, we define a property $\mathfrak{R}'$ by splitting up the input truth table for $\mathfrak{R}'$ into independent blocks, and accepting iff at least one of the blocks satisfies $\mathfrak{R}$. A simple calculation shows that by choosing the block size appropriately, a property with density $1/\operatorname{poly}(N)$ can be transformed into one with density $1 - 1/\operatorname{poly}(N)$, with the new property being useful against circuits of size $s(cn)$ for some $c > 0$ if the original property is useful against circuits of size $s(n)$. ◀

We also introduce and use a more refined definition of easiness on average, where we separate the complexity of the algorithm solving the problem on average from the complexity of the error set (or more precisely a not too large superset of the error set).

▶ **Definition 13** (Easiness on Average with Bounded Complexity Error Set). Let $\mathfrak{B}$ and $\mathfrak{C}$ be (uniform or non-uniform) complexity classes, and let $\varepsilon : \mathbb{N} \to [0, 1]$ be a success parameter. We say a distributional problem $(L, \mathcal{D})$ is solvable in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε if there is a $\mathfrak{C}$-algorithm A and a $\mathfrak{B}$-algorithm A' such that for each $x \in \{0,1\}^*$, $A(x) = L(x)$ or $A(x) =' ?'$, $A(x) =' ?'$ implies $A'(x) = 1$, and for each $n \in \mathbb{N}$, with probability at least $\varepsilon(n)$ over $x \sim D_n$, $A'(x) = 0$. We say that a language L is in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε if (L, U_n) is in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε. We say that a language L is feasibly in $\mathfrak{C}$ on average with success ε if L is in $\mathfrak{C}$ on average with SIZE(poly)-bounded success ε.

We observe that the above notion is equivalent to the notion in Definition 11 when $\mathfrak{B} = \mathfrak{C}$ and $\mathfrak{C}$ is a standard complexity class such as P or SIZE(poly).

▶ **Proposition 14** (Specialization of the Refined Notion of Average-Case Easiness to the Standard Notion). *Let $\mathfrak{C} =$* P *or* SIZE(poly)*, and $\varepsilon : \mathbb{N} \to [0, 1]$ be a success parameter. L is in $\mathfrak{C}$ on average with $\mathfrak{C}$-bounded success ε iff L is in $\mathfrak{C}$ on average with success ε.*

Proof. The forward direction is immediate. For the backward direction, let A be a $\mathfrak{C}$-algorithm solving L on average with success ε. We define a $\mathfrak{C}$-algorithm A' as follows: $A'(x) = 1$ iff $A(x) =' ?'$. It is easy to see that A and A' satisfy the conditions in Definition 13, showing that L is in $\mathfrak{C}$ on average with $\mathfrak{C}$-bounded success ε. ◀

2.5 Randomness and Pseudorandomness

▶ **Definition 15** (Indistinguishability)**.** Let $\mathfrak{C}$ be a (uniform or non-uniform) complexity class, and $\{D_n\}, \{D'_n\}, n \in \mathbb{N}$ be sequences of distributions such that for each n, D_n and D'_n are supported on $\{0,1\}^n$. Let $\varepsilon : \mathbb{N} \to [0,1]$ be an error parameter. We say $\{D_n\}$ and $\{D'_n\}$ are ε-indistinguishable by $\mathfrak{C}$ if for all $L \in \mathfrak{C}$ and all large enough n,

$$\left| \Pr_{w \sim D_n}[L(w) = 1] - \Pr_{w \sim D'_n}[L(w) = 1] \right| \leq \varepsilon(n).$$

By default, the parameter $\varepsilon(n)$ in the above definition is taken to be $1/n$.

▶ **Definition 16** (Pseudorandom Generators)**.** Let $\ell \colon \mathbb{N} \to \mathbb{N}$, $h \colon \mathbb{N} \to \mathbb{N}$ and $\varepsilon \colon \mathbb{N} \to [0,1]$ be functions, and let $\mathfrak{C}$ be a circuit class. A sequence $\{G_n\}$ of functions $G_n \colon \{0,1\}^{\ell(n)} \to \{0,1\}^n$ is an (ℓ, ε) pseudorandom generator *(PRG)* against $\mathfrak{C}$ if $\{G_n(U_{\ell(n)})\}$ and $\{U_n\}$ are ε-indistinguishable by $\mathfrak{C}$. The pseudorandom generator is called quick if its range is computable in time $2^{O(\ell(n))}$.

We define random reducibility between languages, and random self-reducibility.

▶ **Definition 17** (Random Self-Reducibility)**.** Let $L, L' \subseteq \{0,1\}^*$ be languages. L is said to be randomly reducible to L' if there are constants k, ℓ and polynomial-time computable functions $g \colon \{0,1\}^* \to \{0,1\}^*$ and $h \colon \{0,1\}^* \to \{0,1\}$ satisfying the following conditions:

1. For large enough n, for every $x \in \{0,1\}^n$ and for each $i \in \mathbb{N}$ such that $1 \leq i \leq n^k$, $g(i, x, r) \sim U_n$ when $r \sim U_{n^\ell}$.
2. For large enough n and for every $x \in \{0,1\}^n$:

$$L(x) = h(x, r, L'(g(1, x, r)), L'(g(2, x, r)), \ldots, L'(g(n^k, x, r)))$$

with probability $\geq 1 - 2^{-n}$ when $r \sim U_{n^\ell}$.

We say L is randomly self-reducible if L is randomly reducible to L. Also, given a promise problem $Q = (\Pi_{YES}, \Pi_{NO})$ and a language L, we say Q is randomly reducible to L if the first condition above holds for all large enough strings but the second condition is only required to hold for strings $x \in \Pi_{YES} \cup \Pi_{NO}$.

We also define a new notion of *pseudorandom reducibility* by relaxing the first condition in the above definition.

▶ **Definition 18** (Pseudorandom Self-Reducibility)**.** Let $\mathfrak{C}$ be a complexity class. Let $Q = (\Pi_{YES}, \Pi_{NO})$ be a promise problem, where $\Pi_{YES}, \Pi_{NO} \subseteq \{0,1\}^*$, and let $L \subseteq \{0,1\}^*$ be a language. Q is said to be pseudorandomly reducible to L with respect to $\mathfrak{C}$ if there are constants k, ℓ and polynomial-time computable functions $g \colon \{0,1\}^* \to \{0,1\}^*$ and $h \colon \{0,1\}^* \to \{0,1\}$ satisfying the following conditions:

1. For every sequence $\{(x_n, i_n)\}, n \in \mathbb{N}$ where $x_n \in \{0,1\}^n$ and $1 \leq i_n \leq n^k$ for all $n \in \mathbb{N}$, $\{g(i_n, x_n, U_{n^\ell})\}$ and $\{U_n\}$ are indistinguishable by $\mathfrak{C}$.

2. For large enough n and for every $x \in (\Pi_{YES} \cup \Pi_{NO}) \cap \{0,1\}^n$:

$$L(x) = h(x, r, L(g(1, x, r)), L(g(2, x, r)), \ldots, L(g(n^k, x, r)))$$

with probability $\geq 1 - 2^{-n}$ when $r \sim U_{n^\ell}$.

Q is said to be pseudorandomly self-reducible with respect to $\mathfrak{C}$ if there is a language L consistent with Q such that Q is pseudorandomly reducible to L with respect to $\mathfrak{C}$.

▶ **Definition 19** (Pseudorandom Functions)**.** Let $s : \mathbb{N} \to \mathbb{N}$ be a size function, and let $\mathfrak{C}$ be a complexity class. A pseudo-random function generator (PRFG) with seed length ℓ against $\mathfrak{C}$ is a sequence of functions $\{F_n\}, F_n : \{0,1\}^{\ell(n)} \to \{0,1\}^{2^n}$ such that the function $G_n(z, i)$ giving the i'th bit of $F_n(z)$ is computable in time $\mathrm{poly}(n)$, and the distributions $F_n(U_{\ell(n)})$ and U_{2^n} are $1/2^n$ -indistinguishable by $\mathfrak{C}$.

We note that the definition of pseudorandom functions given here is somewhat different from the standard notion [16], where the distinguisher circuit only gets oracle access to the function it is trying to distinguish from random. Our notion is stronger, and more relevant to the current setting. As shown by [30], the construction of [16] gives pseudorandom functions according to Definition 19, under the assumption that exponentially hard one-way functions exist. We now define this concept.

▶ **Definition 20** (Exponentially Hard One-way Functions)**.** A sequence $\{f_n\}, n \in \mathbb{N}$ of functions, where $f_n : \{0,1\}^n \to \{0,1\}^n$ is said to be a exponentially hard one-way function if $\{f_n\}$ is polynomial-time computable, and there is a constant $\varepsilon > 0$ such that for any sequence $\{C_n\}$ of circuits, where C_n has size at most 2^{n^ε} for large enough n, $\Pr_{y \sim U_n}(f_n(C_n(f_n(y))) = f_n(y)) < 1/2^{n^\varepsilon}$.

▶ **Theorem 21** (PRFG from One-Way Functions; Goldwasser-Goldreich-Micali [16])**.** *If exponentially hard one-way functions exist, then there is a PRFG with seed length* $\mathrm{poly}(n)$ *against* $\mathsf{SIZE}(\mathrm{poly})$.

In a seminal result with significant implications for the provability of circuit lower bounds, Razborov and Rudich [30] showed that the existence of pseudorandom functions is incompatible with the existence of natural properties.

▶ **Theorem 22** (Ruling out Natural Properties using Pseudorandom Function [30])**.** *If exponentially hard one-way functions exist, then there are no* $\mathsf{SIZE}(\mathrm{poly})$*-natural properties useful against* $\mathsf{SIZE}(\mathrm{poly})$.

3 Pseudorandom Self-Reducibility for MCSP

▶ **Theorem 23.** *Suppose exponentially hard one-way functions exist. Let* $s : \mathbb{N} \to \mathbb{N}$ *be a size bound such that* $s(n) = n^{\omega(1)}$*. Then there is a constant* $c > 0$ *such that* $\mathsf{MCSP}[s + n^c, s - n^c]$ *is pseudorandomly self-reducible with respect to* $\mathsf{SIZE}(\mathrm{poly})$.

Proof. Suppose that exponentially hard one-way functions exist. By Theorem 21, there is a PRFG $\{F_n\}$ with seed length $\mathrm{poly}(n)$ against $\mathsf{SIZE}(\mathrm{poly})$. Let c be a constant such that the function G_n corresponding to F_n in Definition 19 is computable in time n^d for some constant $d < c$, and hence by Boolean circuits of size $< n^c$, using the standard simulation of deterministic time by circuit size. We show that there is a pseudorandom reduction from $\mathsf{MCSP}[s + n^c, s - n^c]$ to $\mathsf{MCSP}[s]$. As the language $\mathsf{MCSP}[s]$ is consistent

with the promise problem $\mathsf{MCSP}[s + n^c, s - n^c]$, this yields a pseudorandom self-reduction for $\mathsf{MCSP}[s + n^c, s - n^c]$.

The idea is that the pseudorandom self-reduction is a 1-query reduction which uses its randomness to generate a function pseudorandomly and then XORs the pseudorandom function with the input truth table. It is not hard to see that the output of this process is still pseudorandom; however, the circuit size of the Boolean function corresponding to the output differs from the circuit size of the function corresponding to the input by at most n^c.

We define functions g and h which define a pseudorandom reduction by Definition 18. We choose the constant k to be 0, i.e., this is a pseudorandom reduction which makes just 1 query. Hence we can assume that g has just 2 parameters y and r. Let $y \in \{0,1\}^N$ be an input, where $N = 2^n$. This is the only case we need to argue about – when N is not a power of 2, we can define $g(y, r)$ to be a uniformly random string of length N, and $h(y, r, b) = b$ for any y of length N, random string r and bit b; then, the conditions of Definition 18 are satisfied, as no strings of length N are YES instances of either the promise problem $\mathsf{MCSP}[s + n^c, s - n^c]$ or the language $\mathsf{MCSP}[s]$. In what follows, we assume $N = 2^n$.

The pseudorandom reduction uses a random string r of length $\ell(n)$, where ℓ is the seed length for the PRFG given by Theorem 21 against $\mathsf{SIZE}(\text{poly})$. We define $g(y, r) = y \oplus F_n(r)$. As in the previous paragraph, we define $h(y, r, b) = b$ for any y of length N, random string r of length $\ell(n)$ and bit b.

We argue that this is indeed a valid pseudorandom reduction from $\mathsf{MCSP}[s+n^c, s-n^c]$ to $\mathsf{MCSP}[s]$ for any size function $s(n) = n^{\omega(1)}$. First we need to show that for any sequence $\{y_N\}$ of inputs, where $|y_N| = N$, $g(y_N, U_{\ell(n)})$ and U_N are $1/N$-indistinguishable by $\mathsf{SIZE}(\text{poly})$. Suppose, to the contrary, that there is a sequence of circuits $\{C_N\}$ $1/N$-distinguishing the two distributions, where the size of C_N is $\text{poly}(n)$. Consider the sequence of circuits $\{C'_N\}$, where $C'_N(z) = C_N(z \oplus y_N)$ for any input z of length N. The circuits $\{C'_N\}$ are also of size $\text{poly}(N)$, and it is easy to see that they $1/2^n$-distinguish the distributions $F_n(U_{\ell(n)})$ and U_N, using the fact that $N = 2^n$. But this is a contradiction to the assumption that $\{F_n\}$ is a PRFG against $\mathsf{SIZE}(\text{poly})$.

Next, we need to show that for any y of length N, if y is a YES instance of $\mathsf{MCSP}[s+n^c, s-n^c]$, then $h(y, r, \mathsf{MCSP}[s](g(y, r))) = 1$ with probability at least $1 - 2^{-n}$ over the choice of r, and similarly, if y is a NO instance of $\mathsf{MCSP}[s+n^c, s-n^c]$, then $h(y, r, \mathsf{MCSP}[s](g(y, r))) = 0$ with probability at least $1-2^{-n}$ over the choice of r. In the former case, we have that $\mathtt{fn}(y)$ has circuit complexity at most $s-n^c$, as y is a YES instance. Hence for any r, $f' = \mathtt{fn}(y \oplus F_n(r))$ has circuit complexity at most s, since the function G_n corresponding to F_n is computable by circuits of size less than n^c by assumption, for any r. Therefore $\mathsf{MCSP}[s](g(y, r) = \mathsf{MCSP}[s](\mathtt{tt}(f')) = 1$ with probability 1 over r, and hence $h(y, r, \mathsf{MCSP}[s](g(y, r))) = 1$ with probability 1 over r, as h just outputs its last parameter. A completely analogous argument establishes the claim for an arbitrary NO instance. ◀

▶ **Theorem 24.** *Let $\mathfrak{B}$ and $\mathfrak{C}$ be complexity classes such that $\mathfrak{C}$ contains BPP and is closed under probabilistic polynomial-time disjunctive truth-table reductions, and let $s : \mathbb{N} \to \mathbb{N}$ be a size function. Let $\varepsilon : \mathbb{N} \to [0, 1]$ be a success parameter such that $\varepsilon \geq 2/N$. Suppose there is a pseudorandom function generator against $\mathfrak{B}$. There is a constant $c > 0$ such that if $\mathsf{MCSP}[s]$ is in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε, then $\mathsf{MCSP}[s + n^c, s - n^c]$ is in $\mathfrak{C}$.*

Proof. By assumption, there is a pseudorandom function generator $\{F_n\}$ against $\mathfrak{B}$; without loss of generality, the sequence of functions $\{G_n\}$ corresponding to this generator is computable in size $< n^c$ for some constant c. Let A be the $\mathfrak{C}$-algorithm solving $\mathsf{MCSP}[s]$ on average, and let A' be the $\mathfrak{B}$-algorithm bounding the error set of A. As in the proof of Theorem 23, there is

a pseudorandom reduction from $\mathsf{MCSP}[s+n^c, s-n^c]$ to $\mathsf{MCSP}[s]$ given by $g(y,r) = y \oplus F_n(r)$ and $h(y,r,b) = b$, where $|y| = 2^n$. The key idea is that because $\{F_n\}$ is pseudorandom against $\mathfrak{B}$, A' cannot distinguish the output of the pseudorandom reduction from a purely random string of the same length, and this means that with noticeable probability, the output of the reduction must fall outside the error set. More precisely, let S_N be the set of N-bit inputs on which A' outputs 0, i.e., S_N does not intersect the error set. The density of S_N is at least ε, and this means that the probability that the output of the pseudorandom reduction is in S_N is at least $\varepsilon - 1/N \geq 1/N$ by assumption on ε, for if not, $D(z) = A'(y \oplus z)$ would $1/N$-distinguish U_N from the distribution given by the pseudorandom function generator. By running the reduction $O(N)$ times independently, and each time simulating A on the output and outputting the answer if it is not '?', we get a $\mathfrak{C}$-algorithm for MCSP, using the assumed closure properties of $\mathfrak{C}$. ◀

▶ **Corollary 25.** *Suppose exponentially hard one-way functions exist. Let $s : \mathbb{N} \to \mathbb{N}$ be a size function such that $s(n) = n^{\omega(1)}$. There is a constant c such that if $\mathsf{MCSP}[s]$ is feasibly on average in SZK, then $\mathsf{MCSP}[s+n^c, s-n^c]$ is in SZK.*

Corollary 25 follows from Theorem 24 by using Theorem 21 and the fact that SZK is known to be closed under disjunctive truth-table reductions.

4 De Morgan Formula Lower Bounds for MCSP

In this section, we will prove an unconditional formula lower bound for computing MCSP.

▶ **Theorem 26.** $\mathsf{MCSP}[s]$ *requires a de Morgan formula of size $n^{2-O(1/\sqrt{\log n})}$ for $s(n) = n^{1/2\sqrt{\log n}}$.*

Throughout this section, Γ denotes the shrinkage exponent (i.e. $\Gamma = 2$ [17]). For a Boolean function f, $\mathrm{L}(f)$ denotes the minimum size of a de Morgan formula that computes f. We say that a random restriction $\rho\colon [n] \to \{0,1,*\}$ is p-regular if $\Pr[\rho(x_i) = *] = p$ for any $i \in [n]$.

4.1 A Review of Impagliazzo, Meka and Zuckerman [20]

The proof is based on the results by Impagliazzo, Meka and Zuckerman [20], which show that a pseudorandom restriction is enough to shrink de Morgan formulas:

▶ **Lemma 27** (Impagliazzo, Meka and Zuckerman [20])**.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ and $p^\Gamma \mathrm{L}(f) \geq 1$. Let $\mathcal{R}_{p,l}$ be a distribution of p-regular l-wise random restrictions. Then, $\mathbb{E}_{\rho \sim \mathcal{R}_{p,l}}[\mathrm{L}(f)] \leq O(p^\Gamma \mathrm{L}(f))$ for $l := p^{-\Gamma}$.*

Proof Sketch (based on [25]). The idea is to decompose the formula of size $\mathrm{L}(f)$ into the small subformulas of size at most $l := p^{-\Gamma}$, which enables us to argue that each subformula shrinks under l-wise random restriction. Specifically, by using the fact that a tree of leaf size s can be decomposed into two trees each of which is of size between $s/3$ and $2s/3$ (as in the proof of Spira's theorem [33]), we can decompose a de Morgan formula computing f into subformulas $g_1, \ldots g_m$ such that $l/6 \leq \mathrm{L}(g_i) \leq l$ for each $i \in [m]$. Note that $m \leq 6\mathrm{L}(f)/l$. The variables of these subformulas consist of, in addition to the original variables of f, special variables[1] which refer to the other subtrees. Thus, we have $\mathrm{L}(f|_\rho) \leq \sum_{i=1}^m \mathrm{L}(g_i|_{\rho'})$ for any

[1] In this proof, we do not count the number of special variables in the size $\mathrm{L}(g_i)$ of subformulas.

restriction ρ, where ρ' denotes the restriction such that $\rho'(x_i) = *$ if x_i is a special variable and $\rho'(x_i) = \rho(x_i)$ otherwise. Furthermore, by some appropriate conversion, we may assume that each g_i has at most two special variables.

From now on the goal is, instead of upper bounding $\mathbb{E}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(f|_\rho)]$, to bound $\mathbb{E}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(g_i|_{\rho'})]$. By using a formula for computing the addressing function, we have $\mathrm{L}(g_i|_{\rho'}) \leq \sum_{\sigma\in\{0,1\}^S} (\mathrm{L}(g_i|_{\sigma\rho'}) + |S|)$, where S denotes the set of special variables in g_i and σ denotes an arbitrary assignment to special variables. Once the special variables are removed from g_i by applying a restriction σ, it holds that

$$\mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(g_i|_{\sigma\rho'})] = \mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,\infty}}[\mathrm{L}(g_i|_{\sigma\rho'})] \leq p^\Gamma \mathrm{L}(g_i),$$

where the first equality holds because $g_i|_\sigma$ depends on at most l variables and the second equality holds because of the definition of the shrinkage exponent. To summarize,

$$\begin{aligned}
\mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(f|_\rho)] &\leq \sum_{i=1}^m \mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(g_i|_{\rho'})] \\
&\leq \sum_{i=1}^m \sum_{\sigma\in\{0,1\}^S} \left(\mathop{\mathbb{E}}_{\rho}[\mathrm{L}(g_i|_{\sigma\rho'})] + |S|\right) \\
&\leq \sum_{i=1}^m 4\left(p^\Gamma \mathrm{L}(g_i) + 2\right) \\
&\leq m \cdot 4(p^\Gamma l + 2) \\
&\leq m \cdot 4(p^\Gamma l + 2p^\Gamma l) && \text{(since } l \geq p^{-\Gamma}\text{)} \\
&= 12p^\Gamma ml \\
&\leq 72p^\Gamma \mathrm{L}(f) && \text{(since } m \leq 6\mathrm{L}(f)/l\text{)}\,. \qquad \blacktriangleleft
\end{aligned}$$

In the standard situation, we set $p = n^{-\Omega(1)}$, and hence we require as large independence as $n^{\Omega(1)}$-wise in the previous lemma. However, we can significantly reduce the number of random bits needed to generate pseudorandom restrictions by composing $l = 2^{O(\sqrt{\log n})}$-wise independent random restriction $r = O(\sqrt{\log n})$ times:

▶ **Theorem 28** (Impagliazzo, Meka and Zuckerman [20]). *Let $f\colon \{0,1\}^n \to \{0,1\}$ and $p^\Gamma \mathrm{L}(f) \geq 1$. Let $q = p^{1/r}$ for some nonnegative integer $r \geq 1$. Let $\mathcal{R}^r_{p,l}$ be a distribution of the composition of r independent q-regular l-wise random restrictions. (Hence, the composed random restriction is p-regular.) Then, $\mathbb{E}_{\rho\sim\mathcal{R}^r_{p,l}}[\mathrm{L}(f)] \leq c^r p^\Gamma \mathrm{L}(f)$ for $l := q^{-\Gamma}$ and for some constant c.*

Proof. By induction on $r \geq 1$. Let $c = 72$ be the universal constant in Lemma 27. The base case is exactly the same with Lemma 27. Now let us assume $r > 1$. Fix a composition $\rho_0 \in \mathrm{supp}(\mathcal{R}^{r-1}_{p,l})$ of $r-1$ restrictions. We pick a l-wise independent random restriction $\rho_1 \sim \mathcal{R}_{p,l}$. By applying Lemma 27 for $f|_{\rho_0}$, we obtain

$$\mathop{\mathbb{E}}_{\rho_1\sim\mathcal{R}_{p,l}}[\mathrm{L}(f|_{\rho_0\rho_1})] \leq cq^\Gamma \mathrm{L}(f|_{\rho_0}).$$

By averaging this inequality under distribution $\rho_0 \sim \mathcal{R}^{r-1}_{p,l}$, it holds that

$$\begin{aligned}
\mathop{\mathbb{E}}_{\rho\sim\mathcal{R}^r_{p,l}}[\mathrm{L}(f)] &\leq cq^\Gamma \mathop{\mathbb{E}}_{\rho_0\sim\mathcal{R}^{r-1}_{p,l}}[\mathrm{L}(f|_{\rho_0})] \\
&\leq cq^\Gamma \cdot c^{r-1} q^{\Gamma(r-1)} \mathrm{L}(f) && \text{(by the induction hypothesis)} \\
&= c^r p^\Gamma \mathrm{L}(f). \qquad \blacktriangleleft
\end{aligned}$$

4.2 Proof of Theorem 26

Now we are ready to prove the unconditional formula lower bound for MCSP. First, note that a q-regular l-wise independent random restriction can be sampled by using random $O(l \log n \log \frac{1}{q})$ bits, and that each coordinate of a random restriction can be computed in time a polynomial in the number of random bits (see [11]). Hence, the output of a composition of r q-regular l-wise independent random restrictions has circuit complexity at most $s := \mathrm{poly}(r, l, \log n, \log \frac{1}{q})$ when regarded as a truth table. The circuit complexity s is significantly smaller than the expected number pn of the unrestricted inputs under p-regular random restrictions, for some appropriate parameters. Specifically, let $p := 2^{\sqrt{\log n}}/n$, $q := p^{1/r}$, $l := q^{-\Gamma}$ and $r := C\sqrt{\log n}$ for some large constant C so that $s = \mathrm{poly}(r, p^{-\Gamma/r}, \log n, \log \frac{1}{p}) \leq 2^{\frac{1}{2}\sqrt{\log n}} \ll pn$.

By Theorem 28, we have $\mathbb{E}_{\rho \sim \mathcal{R}^r_{p,l}}[\mathrm{L}(f|_\rho)] \leq c^r p^\Gamma \mathrm{L}(f)$. Hence, the goal is to obtain a lower bound on $\mathbb{E}_{\rho \sim \mathcal{R}^r_{p,l}}[\mathrm{L}(f|_\rho)]$. We claim that a pseudorandom restriction does not shrink the formula for computing MCSP:

▶ **Lemma 29.** *Let $\rho\colon [n] \to \{0, 1, *\}$ be a restriction such that ρ can be computed by a circuit of size s, and let $V := \rho^{-1}(*)$. Then, for $f = \mathsf{MCSP}[s]$, we have $\mathrm{L}(f|_\rho) \geq |V| - O(s \log s)$.*

Proof. Let $V_0 \subset V$ be the set of variables on which $f|_\rho$ does not depend. It suffices to claim that $|V_0| = O(s \log s)$ because $\mathrm{L}(f|_\rho) \geq |V| - |V_0|$.

Indeed, let $\sigma\colon V \to \{0, 1\}$ denote an assignment for variables in V. For $\sigma \equiv 0$, the circuit size of the truth table $\rho \circ \sigma \in \{0, 1\}^n$ is at most s. Hence, $\rho \circ \sigma$ is an YES instance of $\mathsf{MCSP}[s]$. Since $f|_\rho$ does not depend on V_0, any assignment σ such that $V \setminus V_0 \subset \sigma^{-1}(0)$ is also an YES instance of $\mathsf{MCSP}[s]$. The number of such assignments is $2^{|V_0|}$, whereas the number of circuits of size at most s is $s^{O(s)}$. Therefore, we have $2^{|V_0|} \leq 2^{O(s \log s)}$. ◀

We can easily show that $|\rho^{-1}(*)| \geq pn/2$ with probability at least $\frac{1}{2}$ by using pairwise independence of ρ and Chebyshev's inequality. Therefore,

$$\begin{aligned}\mathop{\mathbb{E}}_{\rho \sim \mathcal{R}^r_{p,l}}[\mathrm{L}(f|_\rho)] &\geq \Pr_\rho\left[|\rho^{-1}(*)| \geq \frac{pn}{2}\right] \cdot \mathop{\mathbb{E}}_{\rho \sim \mathcal{R}^r_{p,l}}\left[\mathrm{L}(f|_\rho) \;\middle|\; |\rho^{-1}(*)| \geq \frac{pn}{2}\right] \\ &\geq \frac{1}{2} \cdot \left(\frac{pn}{2} - O(s \log s)\right) \geq \frac{pn}{8},\end{aligned}$$

where the last inequality holds since $O(s \log s) \ll pn$. Thus, $\frac{pn}{8} \leq \mathbb{E}_\rho[\mathrm{L}(f|_\rho)] \leq c^r p^\Gamma \mathrm{L}(f)$ and hence $\mathrm{L}(f) \geq np^{-\Gamma+1}/8c^{-r} = n^2 \cdot 2^{-O(\sqrt{\log n})}$.

5 Average-case $\mathsf{AC}^0[p]$ Lower Bound of MKTP

In this section, we show an unconditional average-case $\mathsf{AC}^0[p]$ circuit lower bound of MKTP. Our result improves a previous result [8] showing a worst-case $\mathsf{AC}^0[p]$ circuit lower bound of MKTP. The whole section is devoted to proving the following result:

▶ **Theorem 30.** *There exists some function $s(n)$ such that $\mathsf{MKTP}[s]$ is not in $\mathsf{AC}^0[p]$ on average with error ϵ, for any prime p and any constant $\epsilon \in (0, 1)$.*

Our proof is based on the techniques of Fefferman, Shaltiel, Umans and Viola [14] They gave a pseudorandom generator against $\mathsf{AC}^0[p]$ that is implicitly computable (i.e. each output bit of the pseudorandom generator is easy to compute, or in other words, the KT-complexity is small). We first focus on the case when $p \neq 2$. In this case, we use the following pseudorandom generator G based on PARITY.

▶ **Definition 31** ([14]). Define $G\colon (\{0,1\}^n)^k \to \{0,1\}^{nk+k}$ as

$$G(x_1,\dots,x_k) := x_1 \cdots x_k \cdot \mathsf{PARITY}(x_1) \cdots \mathsf{PARITY}(x_k)$$

for $(x_1,\dots,x_k) \in (\{0,1\}^n)^k$.

▶ **Lemma 32** (implicit in [14]). *If there is an oracle A that distinguishes G from the uniform distribution with advantage a constant $\epsilon > 0$, then there is an AC^0 circuit C with A-oracle gates such that $C^A(x) = \mathsf{PARITY}(x)$ for any $x \in \{0,1\}^n$.*

Proof Sketch. For completeness, we include a brief proof sketch. They showed that, by using *resamplability* of PARITY, there is an NC^0 circuit C_0 with one A-oracle gate such that $\Pr_{x\sim\mathcal{U}_n}[C_0^A(x) = \mathsf{PARITY}(x)] \geq \frac{1+\epsilon}{2}$ ([14, Lemma 4.5]). By using resamplability again for t independent choices of randomness (for some appropriately chosen t), we obtain circuits $C_1^A,\dots,C_t^A$ each of which approximates PARITY. Now taking the majority vote of these circuits, we can compute PARITY on all inputs. Here, the majority can be implemented by using Approximate Majority [1] in AC^0, because the advantage of approximating PARITY is at least a constant ϵ. As a result, we obtain an AC^0 circuit with A-oracle gates that computes PARITY on all inputs ([14, Proposition 4.21]). ◀

Therefore, it is sufficient to claim that an average-case easiness of $\mathsf{MKTP}[s]$ implies that the pseudorandom generator G can be broken. We first claim that the KT-complexity of any output of the pseudorandom generator G in Lemma 32 is small.

▶ **Claim 33.** $\mathrm{KT}(G(x_1,\dots,x_k)) \leq nk + n \cdot \mathsf{polylog}(n)$ *for any seed* $(x_1,\dots,x_k) \in (\{0,1\}^n)^k$.

Proof. We use a description $d := (x_1,\dots,x_k)$. Given an index $i \in \{1,\dots nk+k\}$ of $G(x_1,\dots,x_k)$, if $i \leq nk$ then output the ith bit of the description d; if $i > nk$ then compute and output $\mathsf{PARITY}(x_{i-nk})$, which takes $O(n)$ steps. A universal machine simulates this computation in time $n \cdot \mathsf{polylog}(n)$. ◀

Therefore, for $k := n^3$, it holds that $\mathrm{KT}(G(x_1,\dots,x_k)) \leq nk + o(k)$ (and thus an MKTP oracle distinguishes G from the uniform distribution).

Now let us assume, towards a contradiction, that there is an $\mathsf{AC}^0[p]$ circuit A_0 that computes $\mathsf{MKTP}[s]$ all but an ϵ fraction of inputs with zero-sided error. We define another circuit A as $A(x) := 1$ if $A_0(x) = 1$ or ?; otherwise $A(x) := 0$. Note that A does not err on yes instances of $\mathsf{MKTP}[s]$. We claim that that A breaks G.

▶ **Claim 34.** *Let $s(n) := n - \sqrt{n}$. The following holds.*

1. $\Pr[A(G(\mathcal{U}_n,\dots,\mathcal{U}_n)) = 1] = 1$.
2. $\Pr[A(\mathcal{U}_{nk+k}) = 1] \leq \epsilon + o(1)$.

Proof.

1. By Claim 33, for any $y = G(x_1,\dots,x_k) \in \{0,1\}^{nk+k}$, we have $\mathrm{KT}(y) = nk + \widetilde{O}(n) \ll n^4 + n^3 - \sqrt{n^4+n^3} = s(nk+k)$; hence, y is an yes instance of $\mathsf{MKTP}[s]$ and $A(y) = 1$.
2. The point is that, under the uniform distribution, there are few yes instances in $\mathsf{MKTP}[s]$. Hence, the algorithm A that solves $\mathsf{MKTP}[s]$ on a $1-\epsilon$ fraction of instances must have a substantial fractions of no instances on which A succeeds. Formally,

$$\begin{aligned}
\Pr_{x\sim\mathcal{U}_{nk+k}}[A(x) = 0] &= \Pr_x[A_0(x) = 0] \\
&= \Pr_x[A_0(x) \neq ? \ \wedge \ x \notin \mathsf{MKTP}[s]] \\
&\geq \Pr_x[A_0(x) \neq ?] - \Pr_x[x \in \mathsf{MKTP}[s]] \\
&\geq 1 - \epsilon - 2^{-\sqrt{nk+k}}.
\end{aligned}$$

◀

In particular, A distinguishes the output of G from the uniform distribution with advantage $1-\epsilon-o(1) \geq \frac{1-\epsilon}{2}$. Now we apply Lemma 32 to obtain an AC^0 circuit C^A with A-oracle gates that solves PARITY. Since $A \in \mathsf{AC}^0[p]$, it shows that $\mathsf{PARITY} \in \mathsf{AC}^0[p]$, which contradicts the lower bounds of Razborov-Smolensky [29, 32] for odd prime p.

When $p = 2$, we use a pseudorandom generator G_{CMD} based on a problem called CMD (connectivity matrix determinant), which was introduced by Ishai and Kushilevitz [21, 22]. For the exact definition of G_{CMD}, the reader is referred to [14]. Here we only need the following property, which easily follows from the fact that CMD is computable in polynomial time.

▶ **Fact 35** (Revised Claim 33). $\mathrm{KT}(G_{\mathsf{CMD}}(x_1, \ldots, x_k)) \leq nk + n^{O(1)}$ *for any seed* $(x_1, \ldots, x_k) \in (\{0,1\}^n)^k$.

▶ **Lemma 36** ([14]). *If there is an oracle A that distinguishes G_{CMD} from the uniform distribution with advantage a constant $\epsilon > 0$, then there is an $\mathsf{AC}^0[2]$ circuit C with A-oracle gates such that $C^A(x) = \mathsf{MAJORITY}(x)$ for any $x \in \{0,1\}^n$.*

Proof Sketch. The problem CMD is resamplable in $\mathsf{AC}^0[2]$ ([14]), and hence as in Lemma 32, CMD can be solved by an $\mathsf{AC}^0[2]$ circuit with A-oracle gates. Since CMD is $\oplus\mathsf{L}$-complete under NC^0 reductions ([22]), $\mathsf{MAJORITY}$ can be also solved by an $\mathsf{AC}^0[2]$ circuit with A-oracle gates. ◀

Combining Fact 35 and Lemma 36, we obtain an $\mathsf{AC}^0[2]$ circuit that solves $\mathsf{MAJORITY}$, which contradicts the lower bound of [29, 32] for the majority function. This completes the proof of Theorem 30.

6 MKTP and Average-case Hardness Conjectures

In this section, we show hardness of MKTP and MCSP under popular hypotheses on average-case hardness of various problems.

6.1 Random 3SAT Hardness of MKTP

Let us consider the distribution of a random 3CNF formula on n variables such that the formula is the conjunction of $m = \Delta n$ clauses sampled from all the possible $2^3\binom{n}{3}$ 3-literal clauses independently and uniformly at random. Given such a formula, Feige's hypothesis states that there is no polynomial-time algorithm that (1) accepts every formula for which all but ϵm clauses are satisfiable (henceforth, call such a formula ϵ-almost satisfiable), and (2) rejects most formulas (i.e. with probability $\frac{1}{2}$ over the choice of a random 3CNF formula).

▶ **Hypothesis 37** (Feige [15]). *For every fixed $\epsilon > 0$ and sufficiently large constant Δ (which are independent of n), there is no polynomial time algorithm that accepts every ϵ-almost satisfiable formula, and rejects most formulas.*

Note that there is a variant of the hypothesis stating that there is no polynomial time algorithm that accepts every *satisfiable* formula and rejects most formulas. This variant is stronger than Hypothesis 37 and may be sensitive to minor model changes (see [15] for more details). Here we refute the weaker hypothesis under MKTP oracle, and hence our result is stronger.

▶ **Theorem 38.** MKTP *is random 3SAT-hard in the sense of* [15]. *That is, there is a polynomial-time algorithm with oracle access to* MKTP *that refutes Hypothesis 37.*

Proof. We construct a many-one reduction from random 3SAT to MKTP. The reduction is simple: given a formula φ, map it to (φ, θ) for some threshold θ chosen later. The idea is that any ϵ-almost satisfiable formula is atypical, and hence it can be described efficiently given an almost satisfying assignment (i.e. the KT-complexity of any ϵ-almost satisfiable formula is small). More specifically, given an assignment x that satisfies all but ϵm clauses of the formula φ, each clause of φ that is satisfied by x has $(2^3-1)\binom{n}{3}$ possibilities; hence, each clause of φ (except for ϵm clauses) is of description length at most $\log 7\binom{n}{3}$. On the other hand, random 3SAT instance are chosen from the space of cardinality $[2^3\binom{n}{3}]^m$, and thus it has KT-complexity roughly $m \log 8\binom{n}{3}$ with high probability. Hence, the MKTP oracle enables us to distinguish ϵ-almost satisfiable formulas from random formulas, by exploiting the difference $m \log 8\binom{n}{3} - m \log 7\binom{n}{3}$ in KT-complexity. Details follow.

Define $\theta := m \log 8\binom{n}{3} - m/20$. We first claim that a random 3SAT formula has KT-complexity at least θ with high probability. Indeed, the number of strings with KT-complexity less than θ is at most 2^θ by simple counting. Thus, since a random 3SAT formula φ is chosen uniformly at random out of the space of cardinarity $[2^3\binom{n}{3}]^m = 2^{\theta + m/20}$, the probability that $\mathrm{KT}(\varphi) < \theta$ is at most $2^{-m/20}$.

The rest of the proof is devoted to proving ϵ-almost satisfiable formula is of low KT-complexity:

▶ **Claim 39.** *For sufficiently small $\epsilon > 0$ and any ϵ-almost satisfiable formula φ, $\mathrm{KT}(\varphi) < \theta$.*

In order to claim that the KT-complexity of φ is small, we need to implement an efficient procedure that, given an index, outputs the clause of φ specified by the index, with random access to a description of φ. We will describe φ by using an ϵ-almost satisfying assignment $x \in \{0,1\}^n$, a subset $S \in \binom{[m]}{\leq \epsilon m}$ of clauses not satisfied by x, $(1-\epsilon)m \log 7\binom{n}{3}$ bits to describe clauses satisfied by x, and $\epsilon m \log \binom{n}{3}$ bits to describe clauses not satisfied by x.

In order to describe each clause of φ efficiently (i.e. in time $\mathsf{polylog}(m)$), there are two issues for which we need ideas from succinct data structures. One is an efficient representation of S. Information theoretically, S can be described in $\log \binom{m}{\epsilon m} \leq \epsilon m \log(em/\epsilon m) = m\epsilon \log(e/\epsilon) < m/100$ bits for sufficiently small $\epsilon > 0$. However, a naive representation of S may not enable us to answer a query $i \stackrel{?}{\in} S$ efficiently; thus, we need the following result.

▶ **Lemma 40** (Brodnik and Munro [13])**.** *There exists a string d_S of length $\log \binom{m}{\epsilon m} + o(\log \binom{m}{\epsilon m})$ and an algorithm that, given random access to d_S and index i, answers a query $i \stackrel{?}{\in} S$ in time $\mathsf{polylog}(m)$.*

The other issue is the use of the ceiling function (c.f. [28, 3]). For each clause satisfied by x, we need $\lceil \log 7\binom{n}{3} \rceil$ bits (if we represent each clause separately), which is not necessarily smaller than $\log 8\binom{n}{3}$ bits. We thus group consecutive $b := 11$ clauses of φ into one block so that each block encodes b clauses by using at most $\lceil b \log 7\binom{n}{3} \rceil$ bits. Since $(\frac{7}{8})^b \leq \frac{1}{4}$, we have $\lceil b \log 7\binom{n}{3} \rceil \leq b \log 8\binom{n}{3} - 1$; thus, we can dispense with 1 bit for each block.

Hence, the KT-complexity of φ is

$$\begin{aligned}
\mathrm{KT}(\varphi) &\leq n + \log \binom{m}{\epsilon m} + o\left(\log \binom{m}{\epsilon m}\right) + \left\lceil \frac{m}{b} \right\rceil \cdot \left\lceil b \log 7\binom{n}{3} \right\rceil + \mathsf{polylog}(m) \\
&\leq \frac{m}{\Delta} + \frac{m}{100} + m \log 8\binom{n}{3} - \frac{m}{b} + o(m) \\
&\leq m \log 8\binom{n}{3} - \frac{m}{20} = \theta
\end{aligned}$$

for sufficiently large Δ and m. ◀

Recently Ryan O'Donnell (personal communication) conjectured a co-nondeterministic version of Feige's hypothesis, i.e., that Hypothesis 37 holds even with respect to co-nondeterministic polynomial-time algorithms. It follows from the proof of Theorem 38 that MKTP is not in coNP under O'Donnell's conjecture. This is the first evidence of any kind that MCSP or MKTP is not in coNP. There has been some speculation about whether MCSP $\in$ SZK, for example this is posed as an open problem in Allender's recent survey [4]. Under a standard derandomization hypothesis, SZK $\subseteq$ NP $\cap$ coNP, hence if MKTP $\in$ SZK, either this hypothesis fails or O'Donnell's conjecture fails.

It should be noted that our proof does not seem to carry over to the case of MCSP. The gap between the KT-complexity of almost satisfiable formulas and random formulas is smaller than $m = o(|\varphi|)$, and it is not clear how to construct a small circuit which simulates the random access machine with additive overhead smaller than m. We leave as an open question to extend Theorem 38 to the case of MCSP.

6.2 Hardness of MCSP under Alekhnovich's hypothesis

While we were not able to prove that MCSP is random 3SAT-hard, we can refute a strong hypothesis about average-case complexity proposed by Alekhnovich [2] under MCSP oracle. He considered a problem of solving linear equations under a certain noise e. Let A be an $m \times n$ matrix over GF(2). Let $D_k(A)$ be the distribution of a random vector $Av + e$, where v is a uniform sample from $\mathrm{GF}(2)^n$ and $e \in \mathrm{GF}(2)^n$ is a uniform sample from the vectors of Hamming weight k (i.e. the number of ones in e is k). Alekhnovich conjectured that there is a matrix such that it is infeasible to distinguish $D_k(A)$ from $D_{k+1}(A)$ efficiently.

▶ **Hypothesis 41** (Alekhnovich [2, Conjecture 4.5])**.** *For every $m(n) = \Theta(n)$, there exists a family of $m(n) \times n$ matrices $\{A_n\}_{n\in\mathbb{N}}$ such that, for every function $k(n)$ which satisfies $n^\epsilon < k(n) < n^{1-\epsilon}$ for some constant $\epsilon > 0$, for every efficient algorithm M, the success probability*

$$|\Pr[M(D_k(A_n)) = 1] - \Pr[M(D_{k+1}(A_n)) = 1]|$$

is negligible.

▶ **Theorem 42.** *There is a polynomial-time algorithm with oracle access to* MCSP *that refutes Hypothesis 41.*

Proof Sketch. Alekhnovich showed that Hypothesis 41 implies the existence of a cryptographic pseudorandom generator ([2, Lemma 4.14]). Now we can construct a pseudorandom function generator as in [16], based on Hypothesis 41. On the other hand, an MCSP oracle can distinguish the output distribution of the pseudorandom function generator from the uniform distribution (see, e.g., [5]). Hence, there is an efficient algorithm that refutes Hypothesis 41 with oracle access to MCSP. ◀

6.3 Planted Clique Hardness of MKTP

Now we move on to planted clique conjectures [23, 26].

Let $G(n, \frac{1}{2})$ denote the distribution of an n-vertex graph whose edges are placed with probability $\frac{1}{2}$ independently (i.e. an Erdős-Rényi random graph). Let $G(n, \frac{1}{2}, k)$ be the distribution of a random graph such that a graph is chosen from $G(n, \frac{1}{2})$ and then a clique of size k is randomly placed in the graph. The decision version of planted clique conjectures states that there is no polynomial time algorithm that distinguishes $G(n, \frac{1}{2}, k)$ from $G(n, \frac{1}{2})$.

On average, there is a clique of size $2\log n$ on $G(n, \frac{1}{2})$, and thus there is a quasipolynomial-time algorithm for solving the planted clique problem by a brute force search. We show that there is a polynomial-time algorithm with oracle access to MKTP that solves the planted clique problem.

▶ **Theorem 43.** *For any $k \geq \mathsf{polylog}(n)$, there is a polynomial-time algorithm with oracle access to* MKTP *that accepts every graph chosen from $G(n, \frac{1}{2}, k)$, and rejects most random graphs chosen from $G(n, \frac{1}{2})$.*

Proof. The idea is the same with the proof of random 3SAT-hardness. As a many-one reduction from the planted clique problem to MKTP, given a random graph G, we map G to (G, θ) for a certain parameter θ.

We first claim that the KT-complexity of most random graphs G chosen from $G(n, \frac{1}{2})$ is large. Indeed, the graph is chosen uniformly at random from the space of cardinarity $2^{\binom{n}{2}}$; thus, the probability that $\mathrm{KT}(G)$ is less than $\binom{n}{2} - k$ is at most $2^{-k} = 1/n^{\omega(1)}$, which is negligible. Define $\theta := \binom{n}{2} - k$.

Next, we claim that the KT-complexity of any graph G with k-clique is less than θ. For this purpose, we present an efficient algorithm that, on input a pair (v, w) of vertices and random access to a description, outputs whether G has an edge between v and w. Let S be a k-clique of G. The description for G consists of the clique S (which is encoded in $k \log n$ bits as the sorted list of vertices in S), and the adjacency matrix of G except for edges connecting vertices in S (which can be encoded in $\binom{n}{2} - \binom{k}{2}$ bits). The algorithm for describing G is as follows: Given the description and a pair (v, w), we first check whether $v \in S$ and $w \in S$ by a binary search. If v and w are in S, then we claim that there is an edge (since S is a clique). Otherwise, we compute an index of the description of an adjacency matrix to which (v, w) corresponds (which can be done in $\mathsf{polylog}(n)$ time), and then output the corresponding bit of the description.

The length of the description is roughly $k \log n + \binom{n}{2} - \binom{k}{2} \ll \theta$, and the time it takes to describe each bit of G is at most $\mathsf{polylog}(n)$. Hence, $\mathrm{KT}(G) < \theta$ for any G with a k-clique. ◀

References

1. Miklós Ajtai and Michael Ben-Or. A theorem on probabilistic constant depth computations. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing (STOC)*, pages 471–474, 1984. doi:10.1145/800057.808715.
2. Michael Alekhnovich. More on average case vs approximation complexity. *Computational Complexity*, 20(4):755–786, 2011. doi:10.1007/s00037-011-0029-x.
3. E. Allender, Joshua Grochow, and Cristopher Moore. Graph isomorphism and circuit size. Technical Report TR15-162, Electronic Colloquium on Computational Complexity, 2015.
4. Eric Allender. The complexity of complexity. In *Computability and Complexity – Essays Dedicated to Rodney G. Downey on the Occasion of His 60th Birthday*, pages 79–94, 2017. doi:10.1007/978-3-319-50062-1_6.
5. Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.
6. Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. In *Symposium on Mathematical Foundations of Computer Science* (MFCS), pages 25–32, 2014. doi:10.1007/978-3-662-44465-8_3.
7. Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008. doi:10.1137/060664537.

8 Eric Allender and Shuichi Hirahara. New insights on the (non)-hardness of circuit minimization and related problems. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:73, 2017. URL: https://eccc.weizmann.ac.il/report/2017/073.

9 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. In *Symposium on Theoretical Aspects of Computer Science* (STACS), volume 30 of *LIPIcs*, pages 21–33, 2015. doi:10.4230/LIPIcs.STACS.2015.21.

10 Eric Allender, Michael Koucky, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded kolmogorov complexity in computational complexity theory. *Journal of Computer and System Sciences*, 77:14–40, 2010.

11 Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms*, 7(4):567–583, 1986. doi:10.1016/0196-6774(86)90019-2.

12 Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. *SIAM Journal on Computing*, 36(4):1119–1159, 2006.

13 Andrej Brodnik and J. Ian Munro. Membership in constant time and almost-minimum space. *SIAM J. Comput.*, 28(5):1627–1640, 1999. doi:10.1137/S0097539795294165.

14 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory of Computing*, 9:809–843, 2013. doi:10.4086/toc.2013.v009a026.

15 Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 534–543, 2002. doi:10.1145/509907.509985.

16 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

17 Johan Håstad. The shrinkage exponent of de morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998. doi:10.1137/S0097539794261556.

18 Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *Conference on Computational Complexity* (CCC), volume 50 of *LIPIcs*, pages 18:1–18:20, 2016. doi:10.4230/LIPIcs.CCC.2016.18.

19 John M. Hitchcock and Aduri Pavan. On the NP-completeness of the minimum circuit size problem. In *Conference on Foundation of Software Technology and Theoretical Computer Science* (FSTTCS), volume 45 of *LIPIcs*, pages 236–245, 2015. doi:10.4230/LIPIcs.FSTTCS.2015.236.

20 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 111–119, 2012. doi:10.1109/FOCS.2012.78.

21 Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 294–304, 2000. doi:10.1109/SFCS.2000.892118.

22 Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *Proceedings of Automata, Languages and Programming, 29th International Colloquium (ICALP)*, pages 244–256, 2002. doi:10.1007/3-540-45465-9_22.

23 Mark Jerrum. Large cliques elude the metropolis process. *Random Structures and Algorithms*, 3:347–359, 1992.

24 Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *Symposium on Theory of Computing* (STOC), pages 73–79, 2000. doi:10.1145/335305.335314.

25 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved average-case lower bounds for De-Morgan formula size. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 588–597, 2013. doi:10.1109/FOCS.2013.69.

26 Ludek Kucera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.

27 Cody D. Murray and Richard Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *Conference on Computational Complexity* (CCC), volume 33 of *LIPIcs*, pages 365–380, 2015. doi:10.4230/LIPIcs.CCC.2015.365.

28 Mihai Patrascu. Succincter. In *In Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 305–313, 2008. doi:10.1109/FOCS.2008.83.

29 Alexander Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987. doi:10.1007/BF01137685.

30 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.

31 Steven Rudich. Super-bits, demi-bits, and NP/qpoly-natural proofs. In *Proceedings of Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 85–93, 1997. doi:10.1007/3-540-63248-4_8.

32 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In Alfred V. Aho, editor, *STOC*, pages 77–82. ACM, 1987. URL: http://dblp.uni-trier.de/db/conf/stoc/stoc87.html#Smolensky87.

33 Philip M Spira. On time-hardware complexity tradeoffs for boolean functions. In *Proceedings of the 4th Hawaii Symposium on System Sciences*, pages 525–527, 1971.

34 Boris Trakhtenbrot. A survey of russian approaches to perebor (brute-force search) algorithms. *IEEE Annals of the History of Computing*, 6(4):384–400, 1984.

35 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.